Application Note

AN2529/D
Rev. 0, 5/2003

Standard Space Vector Modulation – XOR version TPU Function Set (svmStdXor)

By Milan Brejl, Ph.D.

Functional Overview

Standard Space Vector Modulation – XOR version (svmStdXor) is a version of the Standard Space Vector Modulation (svmStd) function that uses two TPU channels to generate one PWM output channel. The TPU channel outputs are connected to an XOR gate whos output is the required PWM signal. See **Figure 1**. An advantage of this solution is the full range 0% to 100% of PWM duty-cycle ratios. There is no *MPW* (minimum pulse width) parameter to limit the edge duty-cycle ratios in this version, unlike in the svmStd. A disadvantage is that the number of assigned TPU channels is doubled.

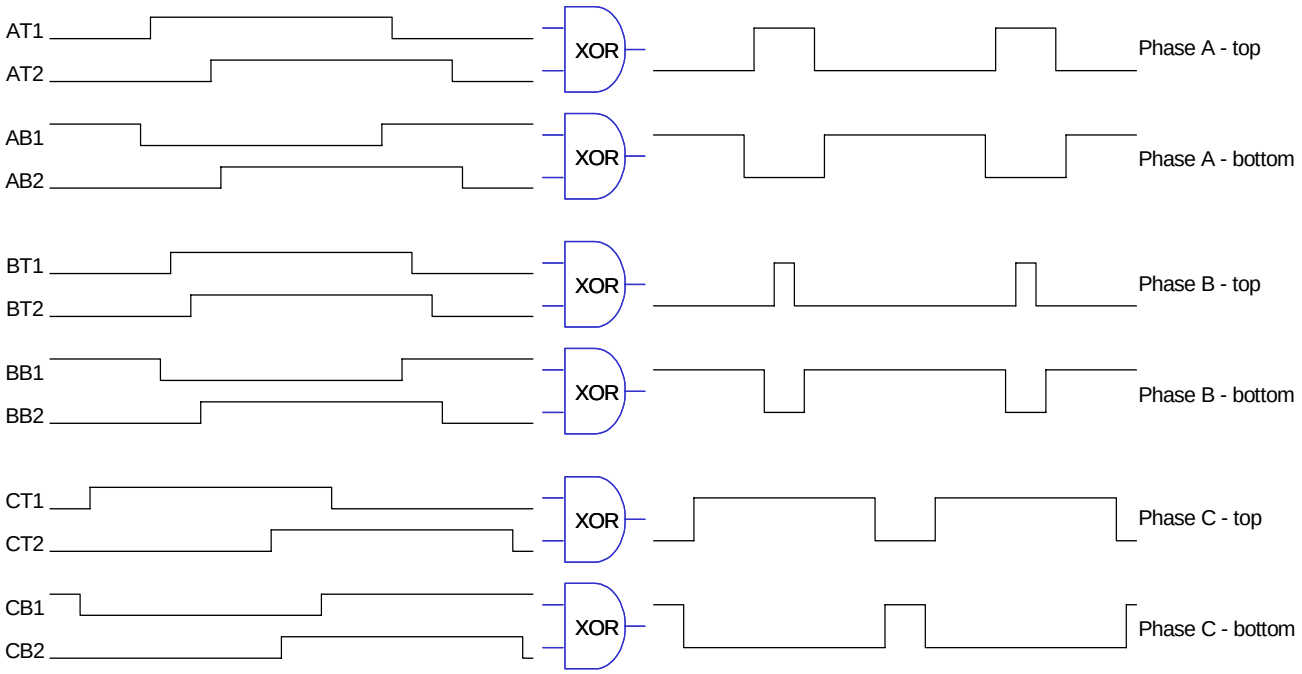


Figure 1. Functionality of XOR version – illustration

The function set consists of 5 TPU functions:

- Standard Space Vector Modulation – XOR version – R channels (svmStdXor_R)
- Standard Space Vector Modulation – XOR version – T channels (svmStdXor_T)
- Synchronization Signal for Standard Space Vector Modulation – XOR version (svmStdXor_sync)
- Resolver Reference Signal for Standard Space Vector Modulation – XOR version (svmStdXor_res)
- Fault Input for Standard Space Vector Modulation – XOR version (svmStdXor_fault)

The svmStdXor_R and svmStdXor_T TPU functions work together to generate 6 pairs of XOR gate inputs. The XOR gate outputs then produce a 6-channel 3-phase center-aligned PWM signal with dead-time between the top and bottom channels. The Synchronization Signal for the svmStdXor function can be used to generate one or more adjustable signals for a wide range of uses, that are synchronized to the PWM, and track changes in the PWM period. The Resolver Reference Signal for the svmStdXor function can be used to generate one or more 50% duty-cycle adjustable signals that are also synchronized to the PWM. The Fault Input for the svmStdXor function is a TPU input function that sets all XOR gate outputs low when the input signal goes low.

Function Set Configuration

None of the TPU functions in the Standard Space Vector Modulation – XOR version TPU function set can be used separately. The svmStdXor_R and svmStdXor_T functions have to be used together. The svmStdXor_R runs on pins AB1, BB1, CB1 – see **Figure 1**. The svmStdXor_T runs on the other pins. One or more channels running Synchronization Signal for svmStdXor as well as Resolver Reference Signals for svmStdXor functions can be added to the svmStdXor_R and svmStdXor_T functions. They can run with different settings on each channel. The function Fault Input for svmStdXor can also be added to the svmStdXor_R and svmStdXor_T functions. It is recommended to use it on channel 15, and to set the hardware option that disables all TPU output pins when the channel 15 input signal is low (DTPU bit = 1). This ensures that the hardware reacts quickly to a pin fault state. Note that it is not only the PWM channels, but all TPU output channels, including the synchronization signals, that are disabled in this configuration.

Table 1 shows the configuration options and restrictions.

Table 1. svmStdXor TPU function set configuration options and restrictions

TPU function	Optional/ Mandatory	How many channels	Assignable channels
svmStdXor_R	mandatory	3	any 3 channels
svmStdXor_T	mandatory	9	any 9 channels
svmStdXor_sync	optional	1 or more	any channels
svmStdXor_res	optional	1 or more	any channels
svmStdXor_fault	optional	1	any, recommended is 15 and DTPU bit set

Table 2 shows an example of configuration.

Table 2. Example of configuration

Channel	TPU function	Priority
0	svmStdXor_T	middle
1	svmStdXor_T	middle
2	svmStdXor_R	middle
3	svmStdXor_T	middle
4	svmStdXor_T	middle
5	svmStdXor_T	middle
6	svmStdXor_R	middle
7	svmStdXor_T	middle
8	svmStdXor_T	middle
9	svmStdXor_T	middle
10	svmStdXor_R	middle
11	svmStdXor_T	middle
13	svmStdXor_sync	low
14	svmStdXor_res	low
15	svmStdXor_fault	high

Table 3 shows the TPU function code sizes.

Table 3. TPU function code sizes

TPU function	Code size
svmStdXor_R	287 μ instructions + 8 entries = 295 long words
svmStdXor_T	3 μ instructions + 8 entries = 11 long words
svmStdXor_sync	26 μ instructions + 8 entries = 34 long words
svmStdXor_res	38 μ instructions + 8 entries = 46 long words
svmStdXor_fault	9 μ instructions + 8 entries = 17 long words

Configuration Order

The CPU configures the TPU as follows.

1. Disables the channels by clearing the two channel priority bits on each channel used (not necessary after reset).
2. Selects the channel functions on all used channels by writing the function numbers to the channel function select bits.
3. Initializes function parameters. The parameters *T*, *prescaler*, *DT*, *SQRT3*, *CPU14* and *sync_presc_addr* must be set before initialization. If an svmStdXor_sync channel or an svmStdXor_res channel is used, then its parameters must also be set before initialization.
4. Issues an HSR (Host Service Request) type %10 to one of the svmStdXor_R channels to initialize all svmStdXor_R and svmStdXor_T channels. Issues an HSR type %10 to the svmStdXor_sync channels, svmStdXor_res channels and svmStdXor_fault channel, if used.
5. Enables servicing by assigning high, middle or low priority to the channel priority bits. All svmStdXor_R and svmStdXor_T channels must be assigned the same priority to ensure correct operation. The CPU must ensure that the svmStdXor_sync or svmStdXor_res channels are initialized after the initialization of the StdDtXor_R and svmStdDtXor_T channels:
   - assign a priority to the StdDtXor_R and svmStdDtXor_T channels to enable their initialization
   - if a Synchronization Signal or a Resolver Reference Signal channel is used, wait until the HSR bits are cleared to indicate that initialization of the StdDtXor_R and svmStdDtXor_T channels has completed and
   - assign a priority to the svmStdXor_sync or svmStdXor_res channels to enable their initialization

NOTE: *A CPU routine that configures the TPU can be generated automatically using the MPC500_Quick_Start Graphical Configuration Tool.*

Detailed Function Description

Standard Space Vector Modulation – XOR version – R channels (svmStdXor_R) and Standard Space Vector Modulation – XOR version – T channels (svmStdXor_T)

The svmStdXor_R and svmStdXor_T TPU functions work together to generate 6 pairs of XOR gate inputs. The XOR gate outputs then produce a 6-channel 3-phase center-aligned PWM signal with dead-time between the top and bottom channels. In order to charge the bootstrap transistors, the PWM signals start to run 1.6ms after their initialization (at 20MHz TCR1 clock). The functions generate signals corresponding to Reference Voltage Vector Amplitude of 0 (50% duty-cycle) until the first reloaded values are processed.

The CPU controls the PWM output by setting the TPU parameters. The Stator Reference Voltage Vector components $u_{\acute{a}}$ and $u_{\hat{a}}$ have to be adjusted during run time. The PWM period *T* and the *prescaler* – the number of PWM periods per reload of new values – are also read at each reload, so these parameters can be changed during run time. Conversely, dead-time (*DT*) is not supposed to be changed during run time. The CPU notifies the TPU that the new reload values are prepared by setting the LD_OK parameter. The TPU notifies the CPU that the reload values have been read and new values can be written by clearing the LD_OK parameter.

The TPU writes the parameter Sector, which indicates the current Stator Reference Voltage Vector position in sector 1 to 6.

The following figures show the input Stator Reference Voltage Vector components $u_{\hat{a}}$ and $u_{\hat{a}}$, corresponding sectors and output PWM signal duty cycle ratios:

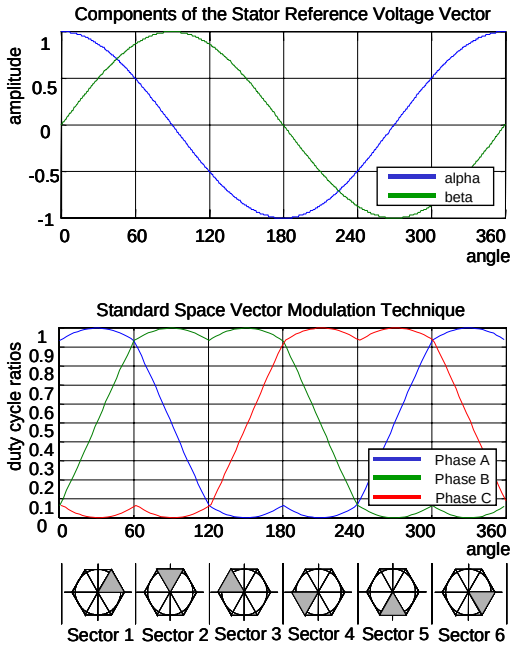


Figure 2. Standard Space Vector Modulation Technique

The following equations describe how the Space Vector Modulation PWM signal high-times ht_A, ht_B, ht_C and transition times t_{trans} of each channel are calculated:

$$U_\beta = T \cdot u_\beta$$

$$U_\alpha = T \cdot u_\alpha$$

$$X = U_\beta$$

$$Y = \frac{U_\beta + U_\alpha \sqrt{3}}{2}$$

$$Z = \frac{U_\beta - U_\alpha \sqrt{3}}{2}$$

	Y < 0			Y >= 0		
	Z < 0	Z >= 0		Z < 0		Z >= 0
		X <= 0	X > 0	X <= 0	X > 0	
Sector:	**V.**	**IV.**	**III.**	**VI.**	**I.**	**II.**

Sector I., IV.: $ht_A = \frac{T+X-Z}{2}$

$ht_B = \frac{T+X+Z}{2} = t_A + Z$

$ht_C = \frac{T-X+Z}{2} = t_B - X$

Sector II., V.: $ht_A = \frac{T+Y-Z}{2}$

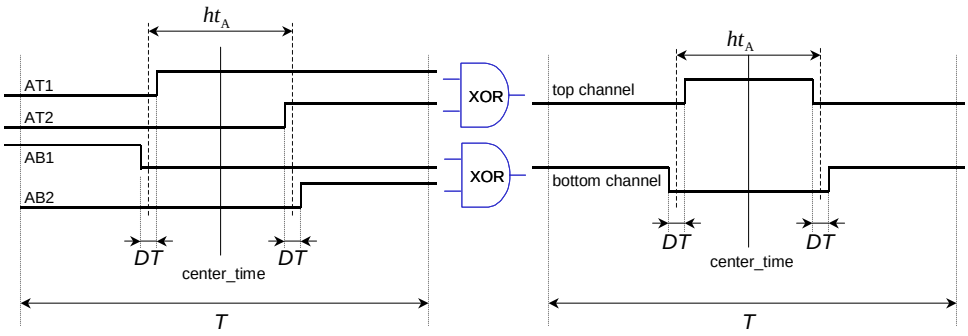
$ht_B = \frac{T+Y+Z}{2} = t_A + Z$

$ht_C = \frac{T-Y-Z}{2} = t_A - Y$

Sector III., VI.: $ht_A = \frac{T-X+Y}{2}$

$ht_B = \frac{T+X-Y}{2} = t_C + X$

$ht_C = \frac{T-X-Y}{2} = t_A - Y$



Phase A:

– T1 channel

$$t_{trans} = \text{center\_time} - \frac{ht_A - DT}{2}$$

– T2 channel

$$t_{trans} = \text{center\_time} + \frac{ht_A - DT}{2}$$

– B1 channel

$$t_{trans} = \text{center\_time} - \frac{ht_A + DT}{2}$$

– B2 channel

$$t_{trans} = \text{center\_time} + \frac{ht_A + DT}{2}$$

Phase B and Phase C similarly with ht_B and ht_C substituted to ht_A.

Host Interface

Written By CPU

Written by both CPU and TPU

Written By TPU

Not Used

Table 4. svmStdXor_T Control Bits

Name	Options
3 2 1 0 Channel Function Select	svmStdXor_T function number (Assigned during assembly the DPTRAM code from library TPU functions)
1 0 Channel Priority	00 – Channel Disabled 01 – Low Priority 10 – Middle Priority 11 – High Priority
1 0 Host Service Bits (HSR)	00 – No Host Service Request 01 – Not used 10 – Not used 11 – Not used
1 0 Host Sequence Bits (HSQ)	xx – Not used
0 Channel Interrupt Enable	x – Not used
0 Channel Interrupt Status	x – Not used

Table 5. svmStdXor_R Control Bits

Name	Options
3 2 1 0 Channel Function Select	svmStdXor_R function number (Assigned during assembly the DPTRAM code from library TPU functions)
1 0 Channel Priority	00 – Channel Disabled 01 – Low Priority 10 – Middle Priority 11 – High Priority
1 0 Host Service Bits (HSR)	00 – No Host Service Request 01 – Not used 10 – Initialization 11 – Stop
1 0 Host Sequence Bits (HSQ)	xx – Not used
0 Channel Interrupt Enable	0 – Channel Interrupt Disabled 1 – Channel Interrupt Enabled
0 Channel Interrupt Status	0 – Interrupt Not Asserted 1 – Interrupt Asserted

TPU function svmStdXor_R generates an interrupt when the current values of *Ualfa*, *Ubeta*, *T* and *prescaler* have been read by the TPU and indicates to the CPU that it can write new variables. The CPU program can either wait for this interrupt to occur, or poll the *LD_OK* bit to check it has cleared. The interrupt is generated at each reload by one of the R channels. The T channels do not generate any interrupts.

Table 6. svmStdXor_T and svmStdXor_R Parameter RAM

Channel	Parameter	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Phase A T1 channel	0	Ttime_AT1															
	1	T_copy															
	2	prsc_copy															
	3	UA															
	4	Ualfa															
	5	Ubeta															
	6																
	7	fault_pinstate															
Phase A T2 channel	0	Ttime_AT2															
	1	min_ht															
	2	max_ht															
	3	UB															
	4	LD_OK															
	5	Sector															
	6																
	7																
Phase A B1 channel	0	htA															
	1	B2_chan_A															
	2	T1_chan_A															
	3	T2_chan_A															
	4	B1a_chan_A															
	5	B1b_chan_A															
	6																
	7																
Phase A B2 channel	0	Ttime_AB2															
	1	state															
	2	center_time															
	3	dec															
	4	T															
	5	prescaler															
	6																
	7																
Phase B T1 channel	0	Ttime_BT1															
	1	UA3															
	2																
	3																
	4	SQRT3															
	5	sync_presc_addr															
	6																
	7																

Table 6. svmStdXor_T and svmStdXor_R Parameter RAM

Channel	Parameter	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Phase B T2 channel	0	Ttime_BT2															
	1																
	2																
	3																
	4	DT															
	5	CPU14															
	6																
	7																
Phase B B1 channel	0	htB															
	1	B2_chan_B															
	2	T1_chan_B															
	3	T2_chan_B															
	4	B1a_chan_B															
	5	B1b_chan_B															
	6																
	7																
Phase B B2 channel	0	Ttime_BB2															
	1																
	2																
	3																
	4																
	5																
	6																
	7																
Phase C T1 channel	0	Ttime_CT1															
	1																
	2																
	3																
	4																
	5																
	6																
	7																
Phase C T2 channel	0	Ttime_CT2															
	1																
	2																
	3																
	4																
	5																
	6																
	7																

Table 6. svmStdXor_T and svmStdXor_R Parameter RAM

Channel	Parameter	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Phase C B1 channel	0	htC															
	1	B2_chan_C															
	2	T1_chan_C															
	3	T2_chan_C															
	4	B1a_chan_C															
	5	B1b_chan_C															
	6																
	7																
Phase C B2 channel	0	Ttime_CB2															
	1																
	2																
	3																
	4																
	5																
	6																
	7																

Table 7. svmStdXor_T and svmStdXor_R parameter description

Parameter	Format	Description
Parameters written by CPU		
Ualfa, Ubeta	16-bit fractional	Stator Reference Voltage Vector components
T	16-bit unsigned integer	PWM period in number of TCR1 TPU cycles
prescaler	16-bit unsigned integer	The number of PWM periods per reload of new values
DT	16-bit unsigned integer	Dead-time in number of TCR1 TPU cycles
CPU14	16-bit unsigned integer	Time of 14 IMB clocks in TCR1 clocks.
SQRT3	16-bit fractional	sqrt(3)/2 = 0.866 = $6EDA constant
sync_presc_addr	8-bit unsigned integer	address of synchronization channel *prescaler* parameter: $X4, where X is synchronization channel number. $0 if no synchronization channel is used.
Parameters written by both TPU and CPU		
LD_OK	1-bit	0 ... CPU can update variables 1 ... TPU can read variables CPU sets 1, TPU sets 0
Parameters written by TPU		
Sector	16-bit unsigned integer	The position of Stator Reference Voltage Vector in a sector. The Sector can be 1, 2, 3, 4, 5 or 6
fault_pinstate	0 or 1	If fault channel is used, state of fault pin: 0 ... low 1 ... high
Other parameters are just for TPU function inner use.		

Table 8. svmStdXor_T State Statistics

State	Max IMB Clock Cycles	RAM Accesses by TPU
ST	2	1
SF	2	0

Table 9. svmStdXor_R State Statistics

State	Max IMB Clock Cycles	RAM Accesses by TPU
INIT	150	35
STOP	166	4
SFR_0	6	1
SFR	44	16
C5	44	15
SFC_0	6	1
SFC	56	11

NOTE: *Execution times do not include the time slot transition time (TST = 10 or 14 IMB clocks)*

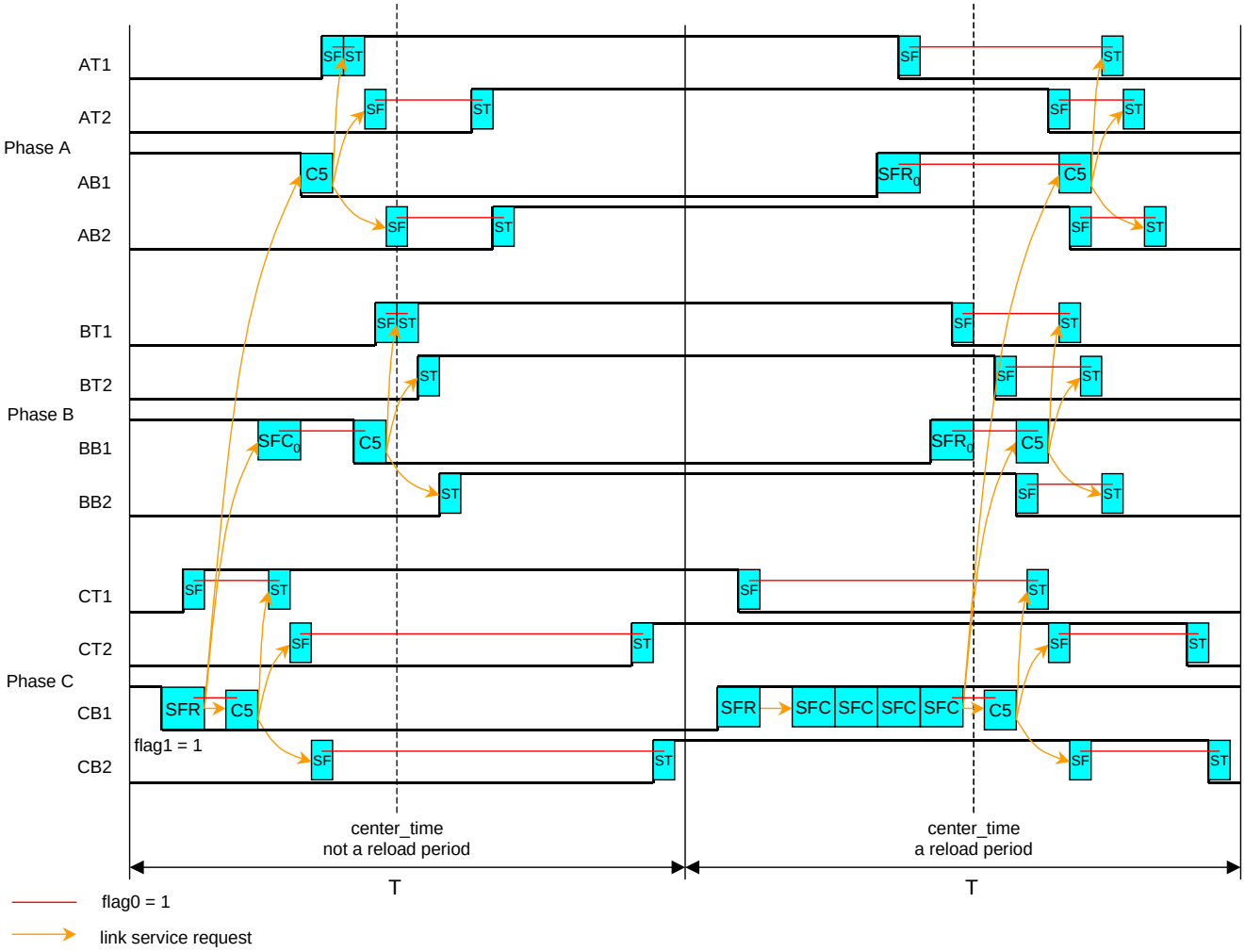


Figure 3. svmStdXor_T and svmStdXor_R timing

NOTE: *The R channel with the momentary earliest transition within the PWM period is marked by a flag1 and runs the SFR and SFC states.*

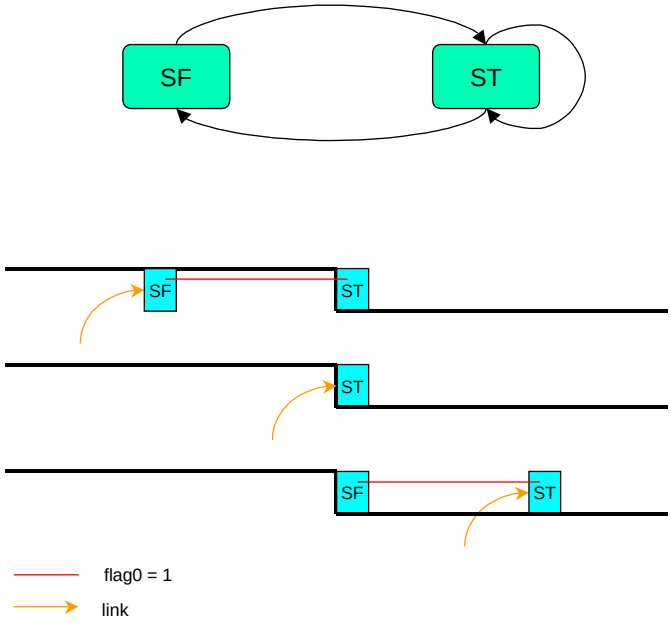


Figure 4. svmStdXor_T state diagram and 3 cases of timing

NOTE: *The case that happens is determined by the time when the link comes.*

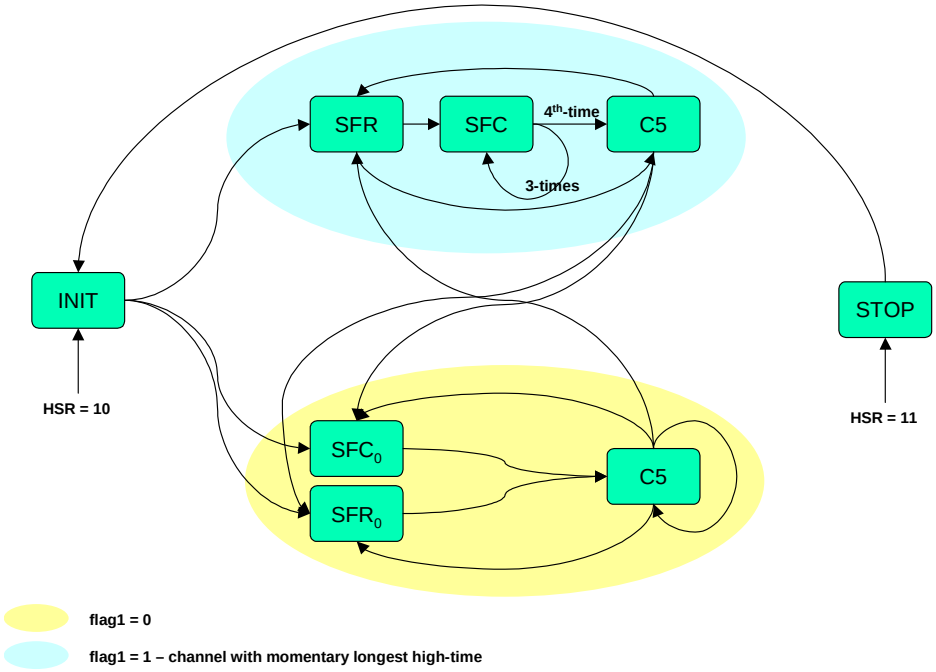


Figure 5. svmStdXor_R state diagram

Synchronization signal for Standard Space Vector Modulation – XOR version (svmStdXor_sync)

The svmStdXor_sync TPU function uses information obtained from StdDtXor_R and svmStdDtXor_T functions, the actual PWM center times and the PWM periods. This allows a signal to be generated, which tracks the changes in the PWM period and is always synchronized with the PWM. The synchronization signal is a positive pulse generated repeatedly after the *prescaler* or *presc_copy* PWM periods (see next paragraph). The low to high transition of the pulse can be adjusted by a parameter, either negative or positive, to go a number of TCR1 TPU cycles before or after the PWM period center time. The pulse width *pw* is another synchronization signal parameter.

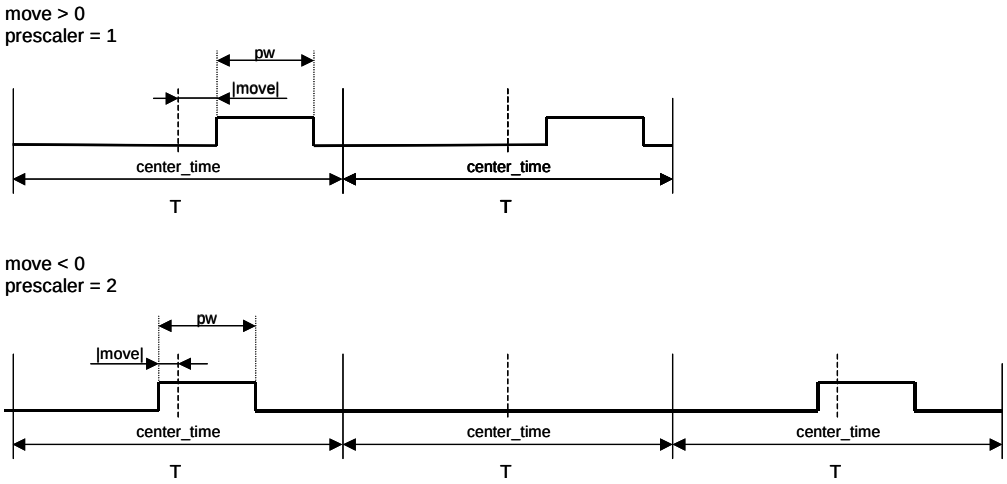


Figure 6. Synchronization signal adjustment examples

Synchronized Change of PWM Prescaler And Synchronization Signal Prescaler

The svmStdXor_sync TPU function actually uses the *presc_copy* parameter instead of the *prescaler* parameter. The *prescaler* parameter holds the prescaler value that is copied to the *presc_copy* by the svmStdXor_bottom function at the time the PWM parameters are reloaded. This ensures that new prescaler values for the PWM signals, as well as the synchronization signal, are applied at the same time. Write the synchronization signal *prescaler* parameter address to the *sync_presc_addr* parameter to enable this mechanism. Write 0 to disable it, and remember to set the synchronization signal *presc_copy* parameter instead of the *prescaler* parameter in this case.

Host Interface

Written By CPU

Written by both CPU and TPU

Written By TPU

Not Used

Table 10. svmStdXor_sync Control Bits

Name	Options
3 2 1 0 Channel Function Select	svmStdXor_sync function number (Assigned during assembly the DPTRAM code from library TPU functions)
1 0 Channel Priority	00 – Channel Disabled 01 – Low Priority 10 –Middle Priority 11 – High Priority
1 0 Host Service Bits (HSR)	00 – No Host Service Request 01 – Not used 10 – Initialization 11 – Not used
1 0 Host Sequence Bits (HSQ)	xx – Not used
0 Channel Interrupt Enable	0 – Channel Interrupt Disabled 1 – Channel Interrupt Enabled
0 Channel Interrupt Status	0 – Interrupt Not Asserted 1 – Interrupt Asserted

TPU function svmStdXor_sync generates an interrupt after each low to high transition.

Table 11. svmStdXor_sync Parameter RAM

Channel	Parameter	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Synchronization channel	0	move															
	1	pw															
	2	prescaler															
	3	presc_copy															
	4	time															
	5	dec															
	6	T_copy															
	7																

Table 12. svmStdXor_sync parameter description

Parameter	Format	Description
Parameters written by CPU		
move	16-bit signed integer	The number of TCR1 TPU cycles to forego (negative) or come after (positive) the PWM period center time
pw	16-bit unsigned integer	Synchronization pulse width in number of TCR1 TPU cycles.
prescaler	16-bit unsigned integer	The number of PWM periods per synchronization pulse – use in case of synchronized prescalers change
presc_copy	16-bit unsigned integer	The number of PWM periods per synchronization pulse – use in case of asynchronized prescalers change
Parameters written by TPU		
Other parameters are just for TPU function inner use.		

Performance

There is one limitation. The absolute value of parameter *move* has to be less than a quarter of the PWM period *T*.

$$|move| < \frac{T}{4}$$

Table 13. svmStdXor_sync State Statistics

State	Max IMB Clock Cycles	RAM Accesses by TPU
INIT	12	5
S1	12	6
S2	8	3
S3	16	7

NOTE: *Execution times do not include the time slot transition time (TST = 10 or 14 IMB clocks)*

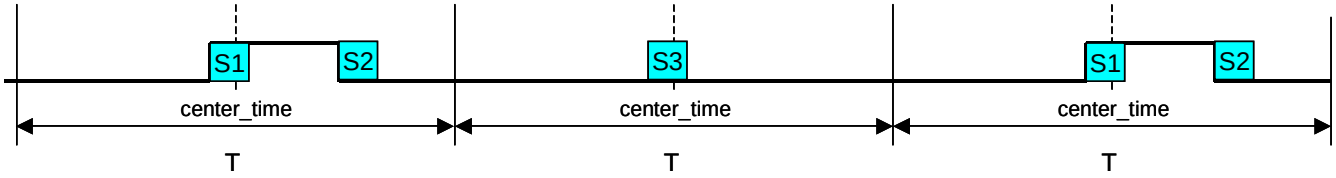


Figure 7. svmStdXor_sync timing

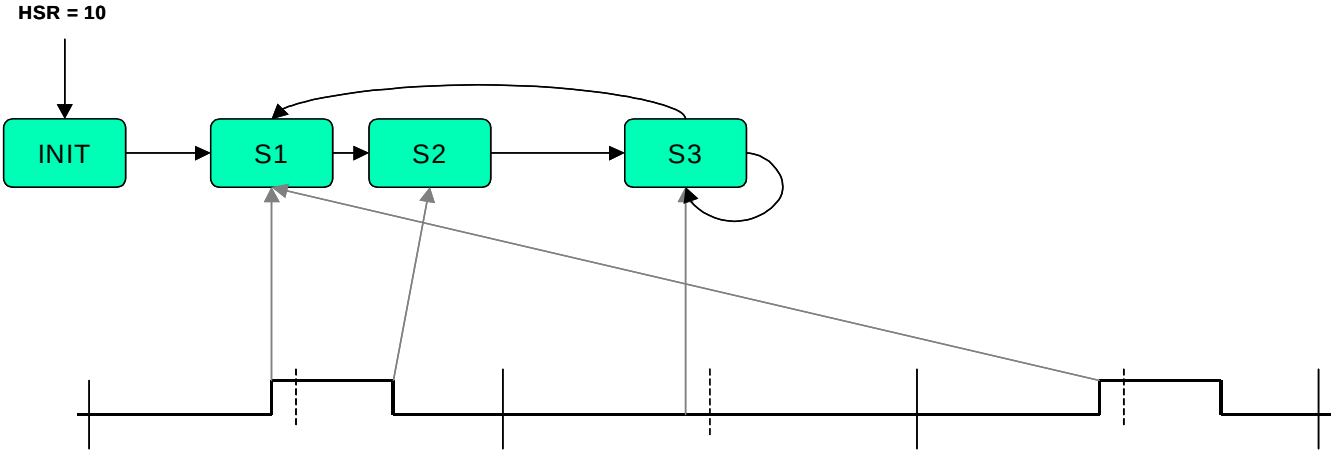


Figure 8. svmStdXor_sync state diagram

Resolver Reference Signal for Standard Space Vector Modulation – XOR version (svmStdXor_res)

The svmStdXor_res TPU function uses information read from the StdDtXor_R and svmStdDtXor_T functions, the actual PWM center times and the PWM periods. This allows a signal to be generated, which tracks the changes of the PWM period and is always synchronized with the PWM. The resolver reference signal is a 50% duty-cycle signal with a period equal to *prescaler* or synchronization channel *presc_copy* PWM periods (see next paragraph). The low to high transition of the pulse can be adjusted by a parameter, either negative or positive, to go a number of TCR1 TPU cycles before or after the PWM period center time.

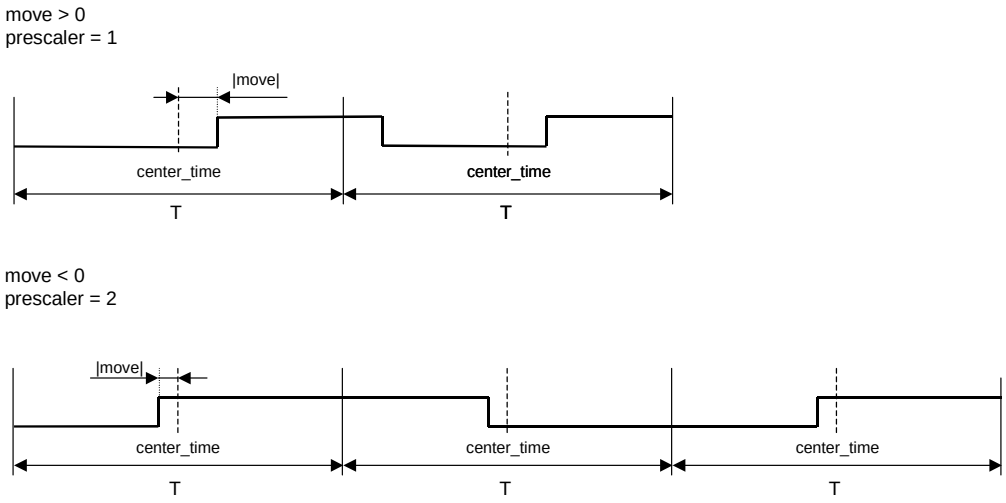


Figure 9. Resolver reference signal adjustment examples

Synchronized Change of PWM Prescaler And Resolver Reference Signals Prescaler

The svmStdXor_res TPU function can inherit the Synchronization Signal prescaler that is synchronously changed with the PWM prescaler. Write the synchronization signals *presc_copy* parameter address to the *presc_addr* parameter to enable this mechanism. Write 0 to disable it, and in this case set the *prescaler* parameter to directly specify prescaler value.

Host Interface

Written By CPU

Written by both CPU and TPU

Written By TPU

Not Used

Table 14. svmStdXor_res Control Bits

Name	Options
3 2 1 0 Channel Function Select	svmStdXor_res function number (Assigned during assembly the DPTRAM code from library TPU functions)
1 0 Channel Priority	00 – Channel Disabled 01 – Low Priority 10 – Middle Priority 11 – High Priority
1 0 Host Service Bits (HSR)	00 – No Host Service Request 01 – Not used 10 – Initialization 11 – Not used
1 0 Host Sequence Bits (HSQ)	xx – Not used
0 Channel Interrupt Enable	x – Not used
0 Channel Interrupt Status	x – Not used

Table 15. svmStdXor_res Parameter RAM

Channel	Parameter	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Resolver	0	move															
	1																
	2	presc_addr															
	3	prescaler															
	4	time															
	5	dec															
	6	T_copy															
	7																

Table 16. svmStdXor_res parameter description

Parameter	Format	Description
Parameters written by CPU		
move	16-bit signed integer	The number of TCR1 TPU cycles to forego (negative) or come after (positive) the PWM period center time
presc_addr	16-bit unsigned integer	$00X6, where X is a number of Synchronization Signal channel, to inherit Sync. channel prescaler or $0000 to enable direct specification of prescaler value in prescaler parameter
prescaler	1, 2, 4, 6, 8, 10, 12, 14, ...	The number of PWM periods per synchronization pulse – use when apresc_addr = 0
Parameters written by TPU		
Other parameters are just for TPU function inner use.		

Performance

There is one limitation. The absolute value of parameter *move* has to be less than a quarter of the PWM period *T*.

$$|move| < \frac{T}{4}$$

Table 17. svmStdXor_res State Statistics

State	Max IMB Clock Cycles	RAM Accesses by TPU
INIT	12	5
S1	26	9
S3	18	7

NOTE: *Execution times do not include the time slot transition time (TST = 10 or 14 IMB clocks)*

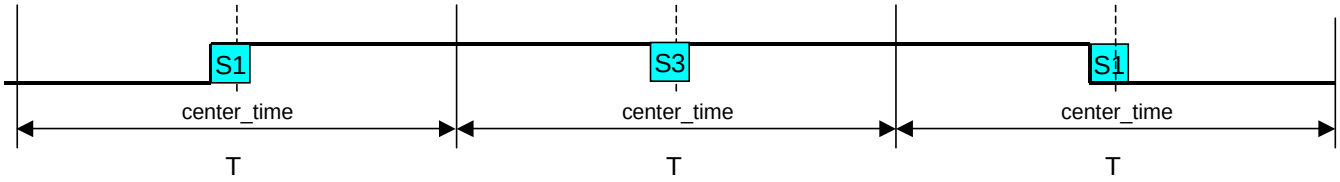


Figure 10. svmStdXor_res timing

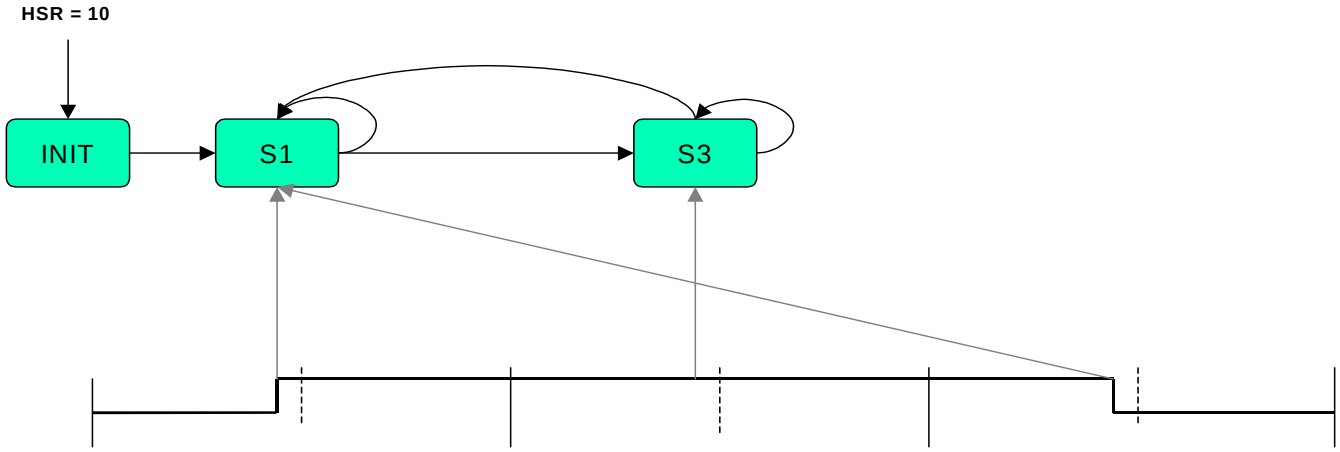


Figure 11. svmStdXor_res state diagram

Fault Input for Standard Space Vector Modulation – XOR version (svmStdXor_fault)

The svmStdXor_fault is an input TPU function that monitors the pin, and if a high to low transition occurs, immediately sets all PWM channels low and cancels all further transitions on them. The PWM channels, as well as the synchronization and resolver reference signal channels (if used), have to be initialized again to start them running.

The function returns the actual pinstate as a value of 0 (low) or 1 (high) in the parameter *fault_pinstate*. The parameter is placed on the AT1 channel to keep the fault channel parameter space free.

Host Interface

Written By CPU

Written by both CPU and TPU

Written By TPU

Not Used

Table 18. svmStdXor_fault Control Bits

Name	Options
3 2 1 0 Channel Function Select	svmStdXor_fault function number (Assigned during assembly the DPTRAM code from library TPU functions)
1 0 Channel Priority	00 – Channel Disabled 01 – Low Priority 10 – Middle Priority 11 – High Priority
1 0 Host Service Bits (HSR)	00 – No Host Service Request 01 – Not used 10 – Initialization 11 – Not used
1 0 Host Sequence Bits (HSQ)	xx – Not used
0 Channel Interrupt Enable	0 – Channel Interrupt Disabled 1 – Channel Interrupt Enabled
0 Channel Interrupt Status	0 – Interrupt Not Asserted 1 – Interrupt Asserted

TPU function svmStdXor_fault generates an interrupt when a high to low transition appears.

Table 19. svmStdXor_fault Parameter RAM

Channel	Parameter	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Fault input	0																
	1																
	2																
	3																
	4																
	5																
	6																
	7																

Table 20. svmStdXor_fault parameter description

Parameter	Format	Description
Parameters written by TPU		
fault_pinstate	0 or 1	State of fault pin: 0 ... low 1 ... high

Performance

Table 21. svmStdXor_fault State Statistics

State	Max IMB Clock Cycles	RAM Accesses by TPU
INIT	8	2
FAULT	172	5
NO_FAULT	4	1

NOTE: *Execution times do not include the time slot transition time (TST = 10 or 14 IMB clocks)*

Figure 12. svmStdXor_fault timing

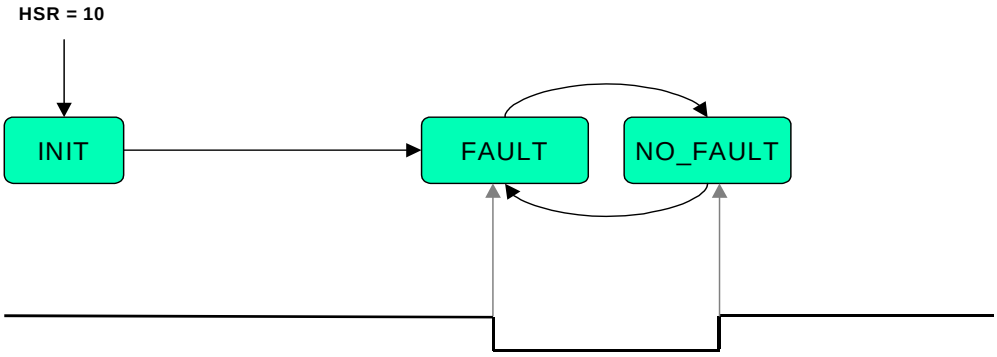


Figure 13. svmStdXor_fault state diagram

[redacted]

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

freescale™
semiconductor

AN2529/D
Rev. 0
5/2003